8

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/818,688	03/26/2001	Youfeng Wu	042390.P10788	9953

7590 05/31/2005

Sanjeet K. Dutta
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
Seventh Floor
12400 Wilshire Boulevard
Los Angeles, CA 90025-1026

EXAMINER	
VO, TED T	
ART UNIT	**PAPER NUMBER**
2192	

DATE MAILED: 05/31/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

Office Action Summary	Application No.	Applicant(s)
	09/818,688	WU, YOUFENG
	Examiner	Art Unit
	Ted T. Vo	2192

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *27 December 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-27* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-8,10-17 and 19-26* is/are rejected.

7)☒ Claim(s) *9, 18 and 27* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *3/07/05*.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the Amendment filed on 12/27/04, responsive to Office Action dated on 9/23/04.

Claim 1, 10, and 19 are amended.

Claims 9, 18, and 27 stand objected as being addressed in Allowable Subject Matter.

Claims 1-8, 10-17, 19-26 have been fully considered by the Examiner, and stand rejected as being addressed below.

The amendment of independent Claims 1, 10, and 19 necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL.** See MPEP § 706.07(a).

Claims 1-27 remain pending in the application.

Response to Amendment

2. Applicant's arguments to Claims 1 and 19, which are rejected under 35 U.S.C. 102(a) as being anticipated by Duesterwald et al., have been fully considered.

Especially, regarding amended limitation of Claims 1 and 19, *wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program edge", Applicants* argue that Duesterwald does not disclose this feature (Remarks page 10, lines 4-8).

Examiner disagrees: The above limitation is a mere definition of phase transition that is computed and defined by Duesterwald as cold path and hot path and its transition (See page Sections 2 and 3). Particularly, refer to the statements, in page 204, right column

"Recall that *freq(p)* denotes the total execution frequency for path *p*. For a set *P* of paths we define the *flow* of *P* as: *freq*(P) = { freq(p)/ p ε P} .
A path *p* is a *hot path* if *freq*(*p*) is greater than some *hot threshold*'",

In page 205, right column

"Once the counter at block A has

> exceeded its threshold, the next executing path is predicted.
> Assume the loop has one or two dominant paths. In such a case,
> NET is statistically likely to predict the correct hot path"

Duesterwald shows transition phase point, "threshold", a changed between hot and cold path.

- Applicants argue that Duesterwald describes software profiling for <u>hot path prediction</u>

Examiner respectfully responds. Edge is only the short path or within adjacent basic blocks such as A → B or A → C, as shown in Figure 1. Mathematically, the "edge" is a particularly case of "path". There is no distinction. For example, if the program shown in Figure 1 contains only the basic block A, B, C only, then it covers the edge profiling.

Applicants argue Duesterwald does not teach **edge profiling**, using hardware and software, direct measuring branch execution frequencies recited in Claims 1 and 19.

Examiner disagrees: Duesterwald discloses these broad limitations and Examiner has already responded the Applicants' arguments in the prior office action date 9/23/04.

- Applicant's arguments to Claims 1 and 19, which are rejected under 35 U.S.C. 102(b) as being anticipated by Conte et al., have been fully considered.

Especially, Applicants argue that Conte's Figure 4 and page 16, passage 3.4 describes transition between blocks along the execution path, not profile the phase transition, and thus Applicants argue Conte does not teach "*wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program edge*".

Examiner respectfully responds, in page 14, left column; particularly, lines 6-45, Conte discloses *detecting profile phase transitions repeatedly, wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program edge*. Conte detecting the phase transition repeatedly in the discussion

> "To limit this explosion, <u>a threshold</u>
> <u>is placed on the execution frequency of a block</u> ('detection'). If
> a block's frequency is <u>below this threshold</u> ('cold spot'), it is not
> considered for trace membership. (This is discussed

in more detail in Section 3.4 below).
There are several methods of recording profile information ('detection').
One method is to insert extra code at
the beginning of each basic block that records the
block id in a buffer ('detection'). This buffer is then parsed into
a WCFG, either periodically during execution, or after
program completes execution ('repeatedly'). One example is
the Spike profiler, which is built into the back end of
GNU CC [16]. A disadvantage is its slowdown, which
is approximately 30 times for Spike.
Another method used by AT&T Global".

Applicant's amendment to Claims 1-8, 10-17, and 19-26, which are rejected under 35 U.S.C. 103(a) over Wu in view of Conte necessitated new ground of rejection.

Especially, Applicants argue no suggestion for combination (Remarks page 13).

Examiner respectfully responds: the prior action has addressed this argument.

Applicant's arguments with respect to Conte does not disclose the newly amended limitation, *wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program* have been considered but are moot in view of the new ground(s) of rejection, or see Examiner's discussion of Claims 1 and 19, which are rejected under 35 U.S.C. 102(b) as being anticipated by Conte et al., above.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. Claims 1 and 19 are rejected under 35 U.S.C. 102(a) as being anticipated by Duesterwald et al., "Software Profiling for Hot Path Prediction: Less is More", ACM 2000.

Given the broadest reasonable interpretation of followed claims in light of the specification:

As per claim 1:

Duesterwald discloses, *"A method, comprising:*

performing repeatedly edge profiling on a program using hardware and software, including directly measuring branch execution frequencies over an entire execution period of the program (Page 202, right column, last paragraph, hardware counters (hardware). Page 205, section 4, Online Prediction Schemes (hardware and software));

detecting profile phase transitions repeatedly, wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program edge (Re: Duesterwald: See page 204, whole section 3. Hot Path prediction; and also see page 210, left column, section 6.1, second paragraph, started with 'Phase changes are implicitly recognized by path prediction scheme...'); *and*

optimizing the program based upon the profile phase transitions and edge profile (Re: Duesterwald: See page 202, section 1, Introduction, particularly, right column, first paragraph, discussing about dynamic compilation systems and dynamic optimizers).

As per Claim 19: Claim 19 recites a computer-readable medium that has the claim limitation corresponding to the functionality of Claim 1. Claim 19 is rejected in the same reason set forth in connecting to the rejection of Claim 1 above.

5. Claims 1 and 19 are rejected under 35 U.S.C. 102(b) as being anticipated by Conte et al., "Using Branch Handling Hardware to Support Profile-Driven Optimization", ACM, 1994.

As per Claim 1:

Given the broadest reasonable interpretation of followed claims in light of the specification:

Conte discloses, *"A method, comprising:*

performing repeatedly edge profiling on a program using hardware and software, including directly measuring branch execution frequencies over an entire execution period of the program (Page 14, left column, line 23-45, a detection of a transition by using the extra code inserted in transition basis block to record the execution along the arc (edge profiling));

detecting profile phase transitions repeatedly, wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program edge (Re: Conte: page 14, left column; particularly, lines 6-45, " threshold is placed on the execution frequency of a block ('detection'). If a block's frequency is below this threshold ('cold spot'), it is not considered for trace membership. (This is discussed in more detail in Section 3.4 below). There are several methods of recording profile information ('detection'). One method is to insert extra code at the beginning of each basic block that records the block id in a buffer ('detection'). This buffer is then parsed into a WCFG, either periodically during execution, or after program completes execution ('repeatedly'). One example is the Spike profiler, which is built into the back end of GNU CC [16]. A disadvantage is its slowdown, which is approximately 30 times for Spike. Another method used by AT&T Global");

optimizing the program based upon the profile phase transitions and edge profile (Re: Conte: Page 12, section 1).

As per Claim 19: Claim 19 recites a computer-readable medium that has the claim limitation corresponding to the functionality of Claim 1. Claim 19 is rejected in the same reason set forth in connecting to the rejection of Claim 1 above.

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> A person shall be entitled to a patent unless –
>
> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claim 1-8, 10-17, and 19-26 are rejected under 35 U.S.C. 103(a) as being unpatentable over by Wu et al., "An Efficient Software-Hardware Collaborative Profiling Technique for Wide-Issue Processors", 1999, in view of Conte et al., "Using Branch Handling Hardware to Support Profile-Driven Optimization".

Given the broadest reasonable interpretation of followed claim in light of the specification:

<u>As per Claim 1</u>:

Wu discloses a method for performing repeatedly edge profiling on a program and detecting profile information given at a branch instruction in a program. The detection is done by insetting memory counters at (See page 1, last two paragraphs, and see Figure 1, page 2). The teaching covers the limitation hereafter:

"performing repeatedly edge profiling on a program using hardware and software, including directly measuring branch execution frequencies over an entire execution period of the program (Re: Wu: See Figure 1, page 2).

detecting profile phase transitions repeatedly, wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program edge; and

optimizing the program (Re: Wu: See page 1, first paragraph of section 1 Introduction, 'runtime profiling and optimization') *based upon the profile phase transitions and edge profile".*

Wu does not particularly address *phase transitions,* but mentions about *block profiling, find frequency execution of basic insert counters at branch instructions* (Re: Wu: See page 1, last two paragraphs).

Conte discloses "*detecting profile phase transitions repeatedly, wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program edge"* (Re: Conte: page 14, left column; particularly, in lines 6-45, "To limit this explosion, a threshold is placed on the execution frequency of a block ('detection'). If a block's frequency is below this threshold ('cold spot'), it is not considered for trace membership (This is discussed in more detail in Section 3.4 below). There are several methods of recording profile information ('detection'). One method is to insert extra code at the beginning of each basic block that records the block id in a buffer ('detection'). This buffer is then parsed into a WCFG, either periodically during execution, or after program completes execution ('repeatedly'). One example is the Spike profiler, which is built into the back end of GNU CC [16]. A disadvantage is its slowdown, which is approximately 30 times for Spike. Another method used by AT&T Global"). Therefore, it would have been obvious to a person of ordinary skill in the art at the time of invention was made to combine the teaching of Wu, *"profile information" detected from branch instructions*, and the teaching of Conte, "*detecting profile phase transitions*". Doing so would extend the basic profiling technique to guide the detection of hotspots more accurate and effective.

<u>As per claim 2</u>:

With regards to the limitation of Claim 2, Wu further discloses using software to insert profile instruction and arrange profile data (Re Wu: See page 6, second paragraph, '...to each of the branch blocks...'), executing that program, and particularly, Wu uses <u>hardware to update the 'profiled information'</u> detected at branch instructions (Re: Wu: See page 6, section 4.2, Profiling hardware, 'At runtime...').

Wu does not particularly address *update profile phase transitions, and signal phase transitions,* but uses the hardware to update profiling information (Re: Wu: See page 6, section 4.2, Profiling hardware, 'At runtime...').

Conte discloses *using hardware* (Re: Conte: Page 17, Figure 4 (b)) *to update profile phase transitions, and signal phase transitions* (Re: Conte: page 14, left column; particularly, lines 6-45).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time of invention was made to include hardware to update '*phase transition*' and '*signal phase transitions'* as disclosed by Conte into the combination of Wu, "profile information" detected from branch instructions, and of Conte, "Calculating the transition change". Doing so would have the support of hardware, and thus would reduce profiling overhead and detect hotspots more accurate and effective.

<u>As per claim 3</u>: Wu further discloses, "*The method of claim 2, wherein using software to insert profiling instructions comprises modifying branch instructions to assign an identifier to one or more profiled edges, and to assign a value to an edge selection field*" (Re: Wu: See page 6, section 4.1.3).

<u>As per claim 4</u>: Wu discloses, "*The method of claim 3, wherein using software to insert profiling instructions further comprises inserting a profile identifier instruction when the profiled edge lacks at least one of a branch instruction;* (see page 6, second paragraph, '...to each of non branch blocks...',) *an initialize profile instruction; and a set offset instruction*" (Re: Wu: See page 4, section 3, Profiling instruction and Registers).

<u>As per claim 5</u>: Wu further discloses, "*The method of claim 2, wherein using hardware comprises translating edge profiling instructions into profile update operations*" (Re: Wu: See page 6, last paragraph, 'three update operations').

<u>As per claim 6</u>: Wu further discloses, "*The method of claim 4, further comprising: loading a profile information register with a base address, an offset value, a trigger-counter, and a flag*" (Re: Wu: See page 7, Figure 3).

<u>As per claim 7</u>: Wu further discloses, "*The method of claim 5, further comprising: intercepting with hardware the profiling instructions; generating a profile update operation; and updating profile counters*" (Re: Wu: See page 2, second bullet, 'update operation to manipulate profile operation').

<u>As per claim 8</u>:

Regarding the limitation of Claim 8, Wu discloses detecting profile information given at a branch instruction in a program. Wu discloses the detection of occurred profile information using a special status register, "profile information register" (Re: Wu: See page 4, section 3), which is dedicated for profiling.

Wu does not disclose, "*profile phase transitions*" and *generating an interrupt signal by the hardware when the profile phase transition occurs*".

Conte discloses the transition change of profiling that uses a counter to update a branch target in a program (Re: Conte: page 14, left column; particularly, lines 6-45, page 16, section 3,4, and page 17, Figure 4). Conte discloses *generating a phase transition interrupt signal* (based on Conte's Figure 4) in discussing handling errors due to the transition difference of the edges (Re: Conte: page 18, section 4.1; particularly see 'Exceptions' in first paragraph of right column), and optimizing based upon transitions and edge profile (Re: Conte: page 12, section 1).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time of invention was made to combine the teachings *detection of profile information using a special status register, "profile information register"* of Wu and *detecting profile changes at the edges and exception handling caused by the transition changes* of Conte. Doing so would take the advantage of hardware supports, and thus would reduce profiling overhead and detect hotspots more accurate and effective.

<u>As per Claim 10</u>:

Wu discloses a system for performing repeatedly edge profiling on a program and detecting profile information given at a branch instruction in a program. The detection is done by insetting memory counters at (See page 1, last two paragraphs, and see Figure 1, page 2). The teaching covers the limitation hereafter:

"*A system, comprising:*

a processor pipeline to generate a profile ID for each profiled edge, and generate profile update operations (Re: Wu: See page 1, last three lines; see page 7, Figure 3)*;*

a profile information register coupled to the processor pipeline (Re: Wu: See page 4, referring to the data structure, 'branch_id ID')*;*

a first logic device to accept the profile update operations and profile ID to generate a memory buffer address (Re: Wu: See page 7, Figure 3)*;*

a profile cache to accept the buffer address connected to the first logic device (Re: Wu: See page 8, Figure 5, 'profile cache')*; and*

Wu does not disclose, *a second logic device coupled to the profile cache configured to generate a phase transition interrupt signal, wherein the system performs edge profiling on a program, detects profile phase transitions repeatedly, wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program edge, and optimizes the program based upon the profile phase transitions*". Wu instead discloses logic devices coupled to the profile cache (Re: Wu: Figure 5, 'Profile operation') configured to generate profiling information signal and to detect profile information repeatedly (Re: Wu: Figure 5, ID → addr → 'Profile operation'). The Wu's system performs edge profiling on a program and optimizes the program based upon the profile information (Re: Wu: page 1, section 1: Introduction, first paragraph, 'runtime profiling and optimization').

Conte discloses "*detects profile phase transitions repeatedly, wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program edge, and optimizes the program based upon the profile phase transitions*".
(Re: Conte: page 14, left column; particularly, lines 6-45, " threshold is placed on the execution frequency of a block ('detection'). If a block's frequency is below this threshold ('cold spot'), it is not considered for trace membership. (This is discussed in more detail in Section 3.4 below). There are several methods of recording profile information ('detection'). One method is to insert extra code at the beginning of each basic block that records the block id in a buffer ('detection'). This buffer is then parsed into a WCFG, either periodically during execution, or after program completes execution ('repeatedly'). One example is the Spike profiler, which is built into the back end of GNU CC [16]. A disadvantage is its slowdown, which is approximately 30 times for Spike. Another method used by AT&T Global");

Therefore, it would have been obvious to a person of ordinary skill in the art at the time of invention was made to combine the teachings *using profile operation that coupled with profile cache* of Wu and *detects profile phase transitions repeatedly, wherein a profile phase transition is an indication that one or more cold program edges have become a corresponding number of hot program edge, and optimizes the program based upon the profile phase transitions* of Conte. Doing so would take the advantage of hardware supports, and thus would reduce profiling overhead and detect hotspots more accurate and effective.

As per claim 11:

With regards to the limitation of Claim 11, Wu further discloses a processor pipeline that executes a program (Re: Wu, page 8, Figure 5), intercepts profiling instructions, and updates profile counters (Re: Wu: Page 4, section 3, "Profiling instructions and registers", and page 8, Figure 5).

Wu does not particularly address *profile phase transitions trigger counters, and signal phase transitions,* but instead, uses special counters for profiling to update profiling information (Re: Wu: Page 4, section 3, "Profiling instructions and registers").

Conte discloses *profile phase transitions, and signal phase transitions* (Re: Conte: in page 14, left column; particularly, lines 6-45).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time of invention was made to take advantage of special counters of Wu for updating '*phase transition*' and '*signal phase transitions'* as disclosed by Conte into the combining teachings of Wu, "profile information" detected from branch instructions, and of Conte, "*profile phase transitions, and signal phase transitions*". Doing so would have the support of hardware counters, and thus would reduce profiling overhead and detect hotspots more accurate and effective.

As per Claim 12: Wu further discloses, "*The system of claim 11, wherein the software inserts edge profiling instructions for modifying branch instructions to assign an identifier to one or more profiled edges, and to assign a value to an edge selection field*" (Re: Wu: See page 6, section 4.1.3).

As per Claim 13: Wu further discloses "*The system of claim 12, wherein the software while inserting edge profiling instructions, also inserts a profile identifier instruction when the profiled edge does not have a branch instruction* (Re: Wu: See page 6, second paragraph, '...to each of non branch blocks...',); *an initialize profile instruction; and a set offset instruction*" (Re: Wu: See page 4, section 3, Profiling instruction and Registers).

As per Claim 14: Wu further discloses, "*The system of claim 11, wherein the processor translates edge profiling instructions into profile update operations*" (Re: Wu: See page 6, last paragraph, 'three update operations').

<u>As per Claim 15</u>: Wu further discloses, "*The system of claim 13, wherein the processor pipeline loads a profile information register with a base address, an offset value, a trigger-counter, and a flag.*" (Re: Wu: See page 7, Figure 3).

<u>As per Claim 16</u>: Wu further discloses, "*The system of claim 14, wherein the processor pipeline: intercepts the profiling instructions; generates a profile update operation; and updates profile counters.*" (Re: Wu: See page 2, second bullet, 'update operation to manipulate profile operation').

<u>As per Claim 17</u>: Regarding the limitation of Claim 17, Wu discloses detecting profile information given at a branch instruction in a program. Wu discloses the detection of occurred profile information using a special status register, "profile information register" (Re: Wu: See page 4, section 3), which is dedicated for profiling.

Wu does not disclose, "*profile phase transitions*" and "*generating an interrupt signal by the hardware when the profile phase transition occurs*".

Conte discloses the transition changes (Re: Conte: Page 14, left column, line 6-45) of profiling that uses a counter to update a branch target in a program a detection of a transition by using the extra code inserted in transition basis block to record the execution along the arc (Page 16, section 3,4, and page 17, Figure 4). Conte discloses *generating a phase transition interrupt signal* (based on Conte's Figure 4) in discussing handling errors due to the transition difference of the edges (Re: Conte: page 18, section 4.1; particularly see 'Exceptions' in first paragraph of right column), and optimizing based upon transitions and edge profile (Re: Conte: page 12, section 1).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time of invention was made to combine the teachings *detection of occurred profile information using a special status register, "profile information register"* of Wu and *detecting profile changes at the edges and exception handling caused by the transition changes* of Conte. Doing so would take the advantage of hardware supports, and thus would reduce profiling overhead and detect hotspots more accurate and effective.

As per Claim 19: Claim 19 recites a computer-readable medium that has the claim limitation corresponding to the functionality of Claim 1. Claim 19 is rejected in the same reason set forth in connecting to the rejection of Claim 1.

As per Claim 20: Claim 20 recites a computer-readable medium that has the claim limitation corresponding to the functionality of Claim 2. Claim 20 is rejected in the same reason set forth in connecting to the rejection of Claim 2.

As per Claim 21: Claim 21 recites a computer-readable medium that has the claim limitation corresponding to the functionality of Claim 3. Claim 21 is rejected in the same reason set forth in connecting to the rejection of Claim 3.

As per Claim 22: Claim 22 recites a computer-readable medium that has the claim limitation corresponding to the functionality of Claim 4. Claim 22 is rejected in the same reason set forth in connecting to the rejection of Claim 4.

As per Claim 23: Claim 23 recites a computer-readable medium that has the claim limitation corresponding to the functionality of Claim 5. Claim 23 is rejected in the same reason set forth in connecting to the rejection of Claim 5.

As per Claim 24: Claim 24 recites a computer-readable medium that has the claim limitation corresponding to the functionality of Claim 6. Claim 24 is rejected in the same reason set forth in connecting to the rejection of Claim 6.

As per Claim 25: Claim 25 recites a computer-readable medium that has the claim limitation corresponding to the functionality of Claim 7. Claim 25 is rejected in the same reason set forth in connecting to the rejection of Claim 7.

As per Claim 26: Claim 26 recites a computer-readable medium that has the claim limitation corresponding to the functionality of Claim 8. Claim 26 is rejected in the same reason set forth in connecting to the rejection of Claim 8.

Allowable Subject Matter

8. Claims 9, 18, and 27 remain objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims and filing terminal disclaimer.

Conclusion

9. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (571) 272-3706. The examiner can normally be reached on 8:00AM to 5:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3694. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

[illegible]

Ted T. Vo
Primary Examiner
Art Unit 2192
May 25, 2005